CS 15-751: A Theorist's Toolkit

Spring 2022 • Prof. Ryan O'Donnell

Notes by Fan Pu Zeng

Contents

Lecture 5 — February 1, 2022 **2**
1 Chernoff Bounds 2

Lecture 11 — February 22, 2022 **3**
1 Error-Correcting Codes 3
2 Linear Codes 4

Lecture 13 — March 1, 2022 **6**
1 Spectral Graph Theory I 6

Lecture 15 — March 15, 2022 **9**
1 Spectral Graph Theory 9

Lecture 17 — March 22, 2022 **11**
1 Linear Programming I 11

Lecture 24 — April 19, 2022 **14**
1 Information Theory 14
2 Entropy 14

Lecture 26 — April 26, 2022 **16**
1 Hardness Assumptions 16
2 LPN: Learning Parity with Noise 16

Lecture 5 — February 1, 2022

Prof. Ryan O'Donnell *Scribe: Fan Pu Zeng*

1 Chernoff Bounds

Let H be the number of heads in n fair coin flips.

Recall the Berry–Esseen theorem. It implies that

$$\mathbb{P}\Big[H \geq \frac{n}{2} + t \cdot \frac{\sqrt{n}}{2}\Big] \approx \overline{\Phi}(t),$$

where $\overline{\Phi}(t)$ is the probability that a standard Gaussian exceeds t.

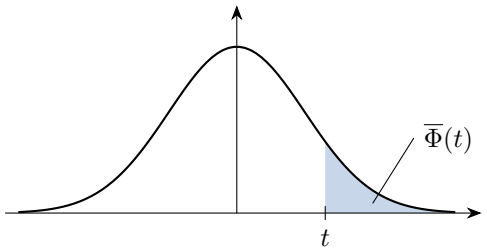

Figure 1: The Gaussian density; the shaded tail to the right of t has area $\overline{\Phi}(t)$.

Writing ϕ for the Gaussian p.d.f., for $t \geq 1$ we have

$$\overline{\Phi}(t) \sim \frac{\phi(t)}{t} \leq \phi(t) \leq e^{-t^2/2}.$$

Lecture 11 — February 22, 2022

Prof. Ryan O'Donnell *Scribe: Fan Pu Zeng*

1 Error-Correcting Codes

Definition (Error-correcting code)**.** An *error-correcting code* (ECC) is an injective map

$$\mathrm{Enc}\colon \Sigma^k \to \Sigma^n.$$

- Σ is the *alphabet*, and $q := |\Sigma|$; often $q = 2$.
- Σ^k is the *message space*, and k is the *message length* (or *dimension*).
- n is the *(block) length*.
- $C := \mathrm{range}(\mathrm{Enc}) \subseteq \Sigma^n$ is *the code*.
- k/n is the *rate*. We want it to be large.

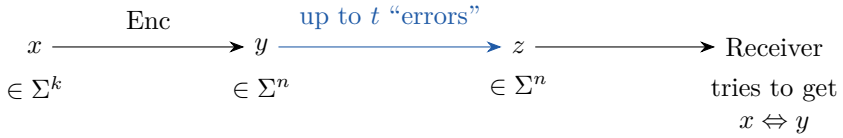

Figure 2: A message x is encoded as y, which is corrupted in up to t positions to z; the receiver tries to recover x (equivalently, y).

Definition (Hamming distance)**.** The *Hamming distance* between $y, z \in \Sigma^n$ is

$$\Delta(y, z) = \#\{i : y_i \neq z_i\}.$$

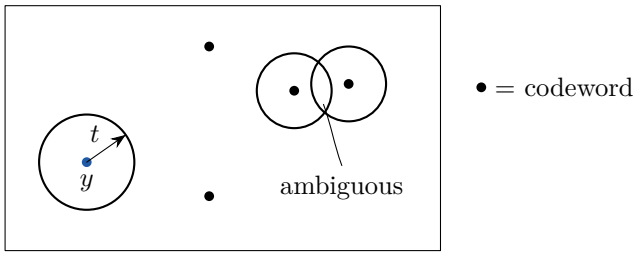

Figure 3: Codewords in Σ^n with radius-t Hamming balls; where two balls overlap, a received word is ambiguous.

We want t to be less than half the minimum distance between codewords.

Definition (Minimum distance)**.** The *minimum distance* of C is

$$d = \min_{\substack{y,y' \in C \\ y \neq y'}} \Delta(y, y').$$

Fact 1. "Unique decoding" is always possible provided $t < d/2$.

The parameters d and k are in conflict with each other.

Idea: choose C randomly?

+ Great rate vs. minimum distance tradeoff.
− Lacks efficient encoding and decoding.

2 Linear Codes

Definition (Linear code)**.** A *linear code* has alphabet $\Sigma = \mathbb{F}_q$, a finite field, and

$$\text{Enc}\colon \mathbb{F}_q^k \to \mathbb{F}_q^n$$

is a linear transformation

$$x \mapsto xG,$$

where G is the $k \times n$ *generator matrix*.

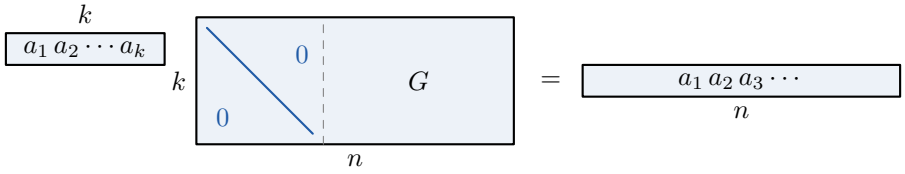

Figure 4: Encoding $x \mapsto xG$. When the left $k \times k$ part of G is the identity (a *systematic code*), the codeword starts with the message itself.

So a codeword is a linear combination of the rows of G, and $C = \text{rowspan}(G)$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

Notation. An $[n, k, d]_q$ code is a linear code in $\mathbb{F}_q^n$ with message length k and minimum distance d.

2.1 The dual code

For a k-dimensional subspace C, define

$$C^\perp = \{w \in \mathbb{F}_q^n : w \cdot y = 0 \ \ \forall y \in C\},$$

where $w \cdot y = \sum_i w_i y_i$ is computed in $\mathbb{F}_q$.

Remark 1 (Exercise)**.** $C^\perp$ is a subspace of $\mathbb{F}_q^n$ (it is closed under linear combinations) of dimension $n-k$, and $(C^\perp)^\perp = C$.

Hence

$$z \in C \iff z \cdot w = 0 \quad \forall w \in C^\perp,$$

which can be used to test whether z is in the code.

Write $C^\perp = \text{rowspan}(H)$. Then $C^\perp$ is an $[n, n-k]_q$ code (the *dual code*), with

$$\text{Enc}^\perp \colon \mathbb{F}_q^{n-k} \to \mathbb{F}_q^n, \qquad x \mapsto xH.$$

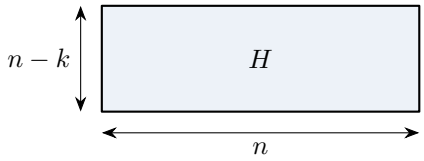

Figure 5: The $(n-k) \times n$ matrix H, whose rows span $C^\perp$.

So

$$z \in C \iff Hz = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

2.2 Minimum distance of a linear code

Claim 1. For a linear code C, $d(C)$ is the minimum Hamming weight of a nonzero codeword.

Proof. Let $y \neq y'$ be in C. Then

$$\Delta(y, y') = \text{Hamming weight}(y - y'),$$

and $y - y' \in C$ because C is linear. ■

Consequently, $d(C)$ is the minimum number of columns of H that are linearly dependent.

2.3 Hamming codes

Definition (Hamming code)**.** The *Hamming code* is defined by having the parity check matrix H whose columns are every vector in $\mathbb{F}_2^r$ except the 0 vector. It is a $[2^r - 1,\ 2^r - 1 - r,\ 3]_2$ code: the length is $n = 2^r - 1$, the message length is $k = 2^r - 1 - r \approx n - \log_2 n$, and the minimum distance is 3.

Suppose the received word is $z = y + e_i$ for a codeword y. Then

$$Hz = H(y + e_i) = Hy + He_i = He_i,$$

since $Hy = 0$. So Hz is the i-th column of H; the columns are distinct, so this identifies the error position i.

Lecture 13 — March 1, 2022

Prof. Ryan O'Donnell *Scribe: Fan Pu Zeng*

1 Spectral Graph Theory I

Throughout, $G = (V, E)$ is an *undirected* graph, which is:

- finite;
- allowed to have multiple edges and self-loops;
- without vertices of degree 0.

Spectral graph theory: understanding properties of graphs using eigenvalues!

1.1 Functions on vertices

We consider functions $f : V \to \mathbb{R}$. For example, $f = 1_S$ for a set $S \subseteq V$, where

$$f(v) = \begin{cases} 1 & \text{if } v \in S, \\ 0 & \text{if } v \notin S. \end{cases}$$

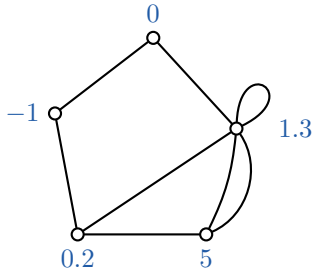

Figure 6: A graph (with a self-loop and a double edge) and a function $f : V \to \mathbb{R}$, written next to each vertex.

We can also think of f as a vector. With $n = |V|$,

$$f = \begin{bmatrix} f(v_1) \\ f(v_2) \\ \vdots \\ f(v_n) \end{bmatrix} \begin{matrix} \leftarrow v_1 \\ \leftarrow v_2 \\ \vdots \\ \leftarrow v_n \end{matrix}$$

All such f form a vector space $\cong \mathbb{R}^n$.

1.2 The quadratic form for G

Definition (Quadratic form)**.** The *quadratic form* of G (the notation is non-standard) is

$$\mathcal{E}[f] := \frac{1}{2} \mathop{\mathbb{E}}_{u \sim v} \big[(f(u) - f(v))^2\big],$$

where $u \sim v$ denotes a uniformly random edge $(u, v) \in E$.

We think of $\mathcal{E}[f]$ as the "local variance" of f. It satisfies facts that resemble facts about variance:

- $\mathcal{E}[f] \geq 0$;
- $\mathcal{E}[cf] = c^2 \, \mathcal{E}[f]$;
- $\mathcal{E}[f + c] = \mathcal{E}[f]$.

If $f = 1_S$, then

$$\begin{aligned}
\mathcal{E}[f] &= \frac{1}{2} \mathop{\mathbb{E}}_{u \sim v \text{ random edge}} \big[(1_S(u) - 1_S(v))^2\big] \\
&= \frac{1}{2} \mathop{\mathbb{P}}_{u \sim v} \big[1_S(u) \neq 1_S(v)\big] \\
&= \mathop{\mathbb{P}}_{u \sim v} \big[u \in S,\ v \notin S\big] \\
&= \mathop{\mathbb{P}}_{\text{random directed edge}} \big[\text{stepping out of } S\big].
\end{aligned}$$

1.3 Random vertices and the stationary distribution

To choose a random vertex:

- choose $u \sim v$ at random;
- output u (or v; this results in the same distribution).

Call this distribution on vertices π.

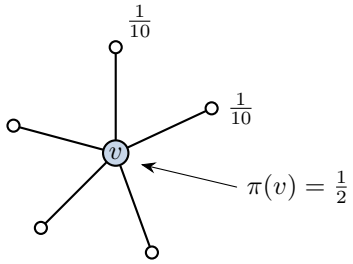

Figure 7: A star with five edges: the center v has $\pi(v) = 5/10 = 1/2$ and each leaf has probability $1/10$.

Fact 2. $\pi(u)$ is proportional to $\deg(u)$:

$$\pi(u) = \frac{\deg(u)}{2|E|}.$$

(This is uniform if the graph is regular.)

Fact 3. Suppose we

- draw $u \sim \pi$;
- let $v \sim u$ be a uniformly random neighbor of u;
- output (u, v).

This generates a uniformly random edge.

Corollary 1. *Let $t \in \mathbb{N}$. Draw $u \sim \pi$, take t random walk steps $u \overset{t}{\rightsquigarrow} v$, and output v. Then the distribution of v is π.*

So π is the *invariant* (or *stationary*) distribution.

Question. Say $u_0 \in V$ is *not* random. Do a simple random walk (SRW) for t steps, $u_0 \overset{t}{\rightsquigarrow} v$. As $t \to \infty$, does the distribution of v tend to π?

Answer. Not if

- G is disconnected, or
- G is bipartite.

CS 15-751: A Theorist's Toolkit Spring 2022

Lecture 15 — March 15, 2022

Prof. Ryan O'Donnell *Scribe: Fan Pu Zeng*

1 Spectral Graph Theory

1.1 Expanding in the eigenbasis of L

Let $\varphi_0, \dots, \varphi_{n-1}$ be orthonormal eigenvectors of L, with eigenvalues $\lambda_0, \dots, \lambda_{n-1}$. Write

$$f = \hat{f}(0)\,\varphi_0 + \hat{f}(1)\,\varphi_1 + \cdots, \tag{$*$}$$

where each $\hat{f}(i)$ is a scalar. Then

$$Lf = \lambda_0 \hat{f}(0)\,\varphi_0 + \lambda_1 \hat{f}(1)\,\varphi_1 + \cdots + \lambda_{n-1}\hat{f}(n-1)\,\varphi_{n-1}.$$

Note that we always have $\lambda_0 = 0$.

That is,

$$\widehat{Lf}(i) = \lambda_i \hat{f}(i).$$

Also, say $g = \hat{g}(0)\,\varphi_0 + \cdots + \hat{g}(n-1)\,\varphi_{n-1}$. Then

$$\begin{aligned}\langle f, g\rangle &= \sum_{i,j} \hat{f}(i)\,\hat{g}(j)\,\langle \varphi_i, \varphi_j\rangle \\ &= \sum_{0\le i\le n-1} \hat{f}(i)\,\hat{g}(i),\end{aligned}$$

since the φ_i are orthonormal. In particular,

$$\mathcal{E}[f] = \langle f, Lf\rangle = \sum_{0\le i\le n-1} \lambda_i \hat{f}(i)^2.$$

1.2 Conductance and the Markov operator

Recall that for $S \subseteq V$, the *conductance* of S is

$$\Phi[S] = \mathop{\mathbb{P}}_{u\sim v}\big[v \notin S \;\big|\; u \in S\big] = \frac{\mathcal{E}[1_S]}{\mathrm{vol}(S)}, \qquad \text{where } \mathrm{vol}(S) = \mathop{\mathbb{E}}_{u\sim\pi}[1_S].$$

The Markov operator K has eigenvectors

$$\mathbf{1} = \varphi_0,\ \dots,\ \varphi_{n-1},$$

with eigenvalues

$$1 = \kappa_0 \ge \kappa_1 \ge \cdots \ge \kappa_{n-1} \ge -1,$$

with equality in the last inequality if G is bipartite.

With $m = |E|$,

$$\pi[u] = \frac{\deg(u)}{2|E|} = \frac{\deg(u)}{2m}.$$

Lecture 17 — March 22, 2022

Prof. Ryan O'Donnell *Scribe: Fan Pu Zeng*

1 Linear Programming I

Linear programming is "the greatest polytime algorithm".

1.1 The problem

Input. A system of linear inequalities in *standard form*, where all constraints are $\geq$:

$$K := \left\{ \begin{array}{rl} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & \geq b_1 \\ & \geq b_2 \\ & \vdots \\ a_{m1}x_1 + \cdots + a_{mn}x_n & \geq b_m \end{array} \right\} \qquad \text{(the } \textit{primal LP}\text{)},$$

where the a_{ij}'s and b_j's are in $\mathbb{Q}$.

Everything lives in $\mathbb{R}^n$. Each constraint defines a half-space, and K is the intersection of the half-spaces.

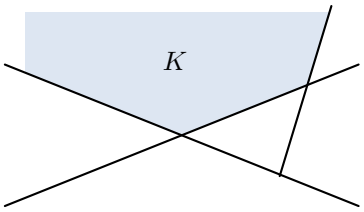

Figure 8: Each constraint is a half-space; K (shaded) is their intersection.

Output. Decide whether $K \neq \emptyset$ or $K = \emptyset$.

- If $K \neq \emptyset$, also output some $x \in K$.
- If $K = \emptyset$, in addition output a "certificate" that $K = \emptyset$.

Theorem 1 (Khachiyan '79)**.** LP $\in$ P.

Theorem 2 (Farkas' Lemma / LP duality)**.** *$K = \emptyset$ if and only if there exist $\lambda_j \geq 0$ such that $(*)$ happens: we can show $0 \geq 1$ by multiplying the constraints by the λ_j and adding them, so that the left-hand sides sum to 0 and the right-hand sides sum to 1.*

1.2 Bit complexity

Let $L = \langle \text{input} \rangle$ be the number of bits in the input. We hope that the answer can be written in $\leq \text{poly}(L)$ bits.

It is not obvious that you can solve linear equations in polynomial time, and that each step takes polynomial time.

Theorem 3.

(1) When $K \neq \emptyset$, there exists a solution $x \in \mathbb{Q}^n$ expressible with $\text{poly}(L)$ bits.

(2) When $K = \emptyset$, there exists a certificate $\lambda \in \mathbb{Q}^m$ expressible with $\text{poly}(L)$ bits.

Proof idea. It suffices to prove (1).

Observation. Given K, the question "do there exist $\lambda_1, \ldots, \lambda_m$ which yield $0 \geq 1$?" is itself an LP problem:

$$\left.\begin{aligned} b_1\lambda_1 + \cdots + b_m\lambda_m &= 1 \\ \lambda_1 \geq 0,\ \lambda_2 \geq 0,\ &\ldots \\ a_{11}\lambda_1 + a_{21}\lambda_2 + \cdots + a_{m1}\lambda_m &= 0 \\ &\vdots \\ a_{1n}\lambda_1 + a_{2n}\lambda_2 + \cdots + a_{mn}\lambda_m &= 0 \end{aligned}\right\} \quad \text{(the } \textit{dual LP}\text{)}.$$

This LP is in *equational form*: all variables are ≥ 0, and all constraints are $=$. Applying (1) to it gives (2). ■

1.3 Proof essence

Suppose $K \neq \emptyset$. Then (and this is actually wrong; see below) there exist vertices, also called extreme points or basic feasible solutions; call one of them x^*.

Then x^* is the intersection of n linearly independent hyperplanes from K (turning $\geq$ into $=$). So x^* is the solution to an $n \times n$ system of equations.

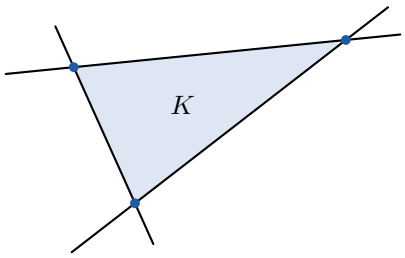

Figure 9: A feasible region K whose vertices (marked) are each the intersection of $n = 2$ of the constraint lines.

Why is this wrong? It is possible for K to be feasible but have no vertices! For example, K could be a single half-space:

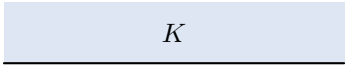

Figure 10: A half-space: feasible, but with no vertices.

Fix: "big box constraints". Add the constraints

$$-B \leq x_i \leq +B \qquad \text{for each } x_i,$$

where B is a big number, but $\langle B \rangle \leq \text{poly}(L)$.

If K is in equational form, then you can add the big box constraints without changing its satisfiability status.

Lecture 24 — April 19, 2022

Prof. Ryan O'Donnell *Scribe: Fan Pu Zeng*

1 Information Theory

Topics:

- entropy;
- mutual information;
- information complexity.

2 Entropy

Let X be a discrete random variable. The *entropy* of X, denoted $H(X)$, is intuitively "the number of random bits in X": the minimum number of random bits needed to generate a draw from X, on average.

Example 1. Let

$$X = \begin{cases} \text{RED} & \text{with probability } 1/2, \\ \text{GREEN} & \text{with probability } 1/4, \\ \text{BLUE} & \text{with probability } 1/8, \\ \text{YELLOW} & \text{with probability } 1/8. \end{cases}$$

We can generate a draw from X by flipping fair coins:

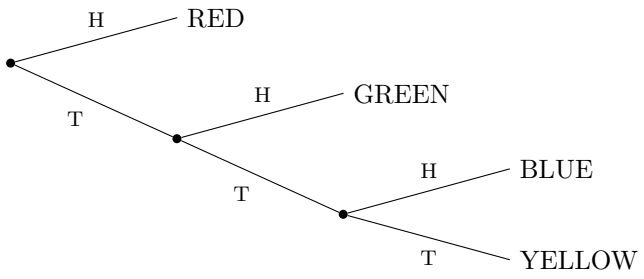

Figure 11: Generating a draw from X with coin flips: heads stops, tails flips again.

The expected number of coin flips is

$$\frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \underbrace{\frac{3}{4}}_{2 \cdot \frac{1}{8} \cdot 3} = 1.75 \text{ bits on average.}$$

Alternatively, $H(X)$ is the number of bits needed to *store* a draw from X, on average, with the best encoding scheme.

CS 15-751: A Theorist's Toolkit Spring 2022

Lecture 26 — April 26, 2022

Prof. Ryan O'Donnell *Scribe: Fan Pu Zeng*

1 Hardness Assumptions

$$\text{Hardness} = \text{Assumption} + \underbrace{\text{Reduction}}_{=\text{ algorithm}}$$

Last time: LWE (learning with errors).

- Does it imply cryptomania?
- Worst-case hardness of GapSVP.
- Hardness of solving "noisy" equations mod a big q.

2 LPN: Learning Parity with Noise

Parameters: n, and $0 < \varepsilon < \frac{1}{2}$.

- A "secret" $s^* \in \mathbb{F}_2^n$ is chosen uniformly at random.
- The algorithm can request (noisy) equations. It gets back

  $$\text{“ } a_1 s_1 + \cdots + a_n s_n = b \text{ ”},$$

  where $a_1, \ldots, a_n \in \mathbb{F}_2$ are uniform, and

  $$b = a_1 s_1^* + \cdots + a_n s_n^* + \eta \pmod 2, \qquad \eta = \begin{cases} 1 & \text{with probability } \varepsilon, \\ 0 & \text{with probability } 1-\varepsilon. \end{cases}$$

2.1 Sparse LPN

Fix $k \geq 3$. The equations are

$$a_1 s_1 + \cdots + a_n s_n = b.$$